Amendments to the Claims:

This listing of claims will replace all prior versions, and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A very long instruction word (VLIW) processing core comprising:

a processing pipeline having N-number of processing paths for processing an instruction comprising N-number ~~or~~ of P-bit instructions appended together to form a VLIW, said N-number of processing paths process said N-number of P-bit instructions in parallel on M-bit data words; and

one or more register files having Q-number of registers, said Q-number of registers being M-bits wide;

wherein one of said Q-number of registers in at least one of said one or more register files is a program counter register which stores a current program counter value.

2. (Original) The processing core as recited in claim 1, wherein one of said Q-number of registers in at least one of said one or more register files is a zero register which always stores zero.

3. (Original) The processing core as recited in claim 1, wherein program jumps are executed by adding a value to the current program counter value stored in the program counter register using a standard add operation.

4. (Original) The processing core as recited in claim 1, wherein memory addresses are calculated by adding a value to the current program counter value stored in the program counter register using a standard add operation.

5. (Original) The processing core as recited in claim 1, wherein program jump tables hold values, which are offset values from the current program counter value.

6. (Original) The processor chip as recited in claim 1, wherein M=64, Q=64, and P=32.

7. (Original) The processing core as recited in claim 1, wherein said Q-number of registers within each of said one or more register files are either private or global registers, and wherein when a value is written to one of said Q-number of said registers which is a global register within one of said plurality of register files, said value is propagated to a corresponding global register in the other of said one or more register files, and wherein when a value is written to one of said Q-number of said registers which is a private register within one of said one or more register files, said value is not propagated to a corresponding register in the other of said one or more register files.

8. (Original) The processing core as recited in claim 7, wherein Q=64, and a 64-bit special register stores bits indicating whether a register in a register file is a private register or a global register, each bit in the 64-bit special register corresponding to one of said registers in said register file.

9. (Original) The processing core as recited in claim 7, wherein said program counter register is a global register.

10. (Original) A processing core comprising:

a processing pipeline having N-number of processing paths, each of said processing paths for processing instructions on M-bit data words; and

one or more register files, each having Q-number of registers, said Q-number of registers being M-bits wide;

wherein one of said Q-number of registers in at least one of said one or more register files is a program counter register which stores a current program counter value; and

wherein said Q-number of registers within each of said one or more register files are either private or global registers, and wherein when a value is written to one of said Q-number of said registers which is a global register within one of said one or more register files, said value is propagated to a corresponding global register in the other of said one or more register files, and wherein when a value is written to one of said Q-number of said registers which is a private register within one of said one or more register files, said value is not propagated to a corresponding register in the other of said one or more register files.

11. (Original) The processing core as recited in claim 10, wherein one of said Q-number of registers in at least one of said one or more register files is a zero register which always stores zero.

12. (Original) The processing core as recited in claim 10, wherein program jumps are executed by adding a value to the current program counter value stored in the program counter register using a standard add operation.

13. (Original) The processing core as recited in claim 10, wherein memory addresses are calculated by adding a value to the current program counter value stored in the program counter register using a standard add operation.

14. (Original) The processing core as recited in claim 10, wherein program jump tables hold values, which are offset values from the current program counter value.

15. (Original) The processing core as recited in claim 10, wherein a processing instruction comprises N-number of P-bit instructions appended together to form a very long instruction word (VLIW), and said N-number of processing paths process N-number of P-bit instructions in parallel.

16. (Original) The processor chip as recited in claim 15, wherein M=64, Q=64, and P=32.

17. (Original) The processing core as recited in claim 16, wherein Q=64, and a 64-bit special register stores bits indicating whether a register in a register file is a private register or a global register, each bit in the 64-bit special register corresponding to one of said registers in said register file.

18. (Original) The processing core as recited in claim 10, wherein said program counter register is a global register.

19. (Currently Amended) In a computer system, a scalable computer processing architecture, comprising:

one or more processor chips, each comprising:

a processing core, including:

a processing pipeline having N-number of processing paths, each of said processing paths for processing instructions on M-bit data words; and

one or more register files, each having Q-number of registers, said Q-number of registers being M-bits wide**, wherein one of the Q-number of registers comprises a program counter register that holds a current program counter value**;

an I/O link configured to communicate with other of said one or more processor chips or with I/O devices;

a communication controller in electrical communication with said processing core and said I/O link;

said communication controller for controlling the exchange of data between a first one of said one or more processor chips and said other of said one or more processor chips;

wherein said computer processing architecture can be scaled larger by connecting together two or more of said processor chips in parallel via said I/O

links of said processor chips, so as to create multiple processing core pipelines which share data therebetween.

20. (Original) The computer processing architecture as recited in claim 19, wherein one of said Q-number of registers in at least one of said one or more register files is a zero register which always stores zero.

21. (Original) The computer processing architecture as recited in claim 19, wherein program jumps are executed by adding a value to the current program counter value stored in the program counter register using a standard add operation.

22. (Original) The processing core as recited in claim 19, wherein memory addresses are calculated by adding a value to the current program counter value stored in the program counter register using a standard add operation.

23. (Original) The computer processing architecture as recited in claim 19, wherein program jump tables hold values, which are offset values from the current program counter value.

24. (Original) The computer processing architecture as recited in claim 19, wherein a processing instruction comprises N-number of P-bit instructions appended together to form a very long instruction word (VLIW), and said N-number of processing paths process N-number of P-bit instructions in parallel.

25. (Original) The computer processing architecture as recited in claim 24, wherein M=64, Q=64, and P=32.

26. (Original) The computer processing architecture as recited in claim 19, wherein said Q-number of registers within each of said one or more register files are either private or global registers, and wherein when a value is written to one of said Q-number of said registers which is a global register within one of said plurality of register files, said value is propagated to a corresponding global register in the other of said one or more register files, and

wherein when a value is written to one of said Q-number of said registers which is a private register within one of said one or more register files, said value is not propagated to a corresponding register in the other of said one or more register files.

27. (Original) The computer processing architecture as recited in claim 26, wherein Q=64, and a 64-bit special register stores bits indicating whether a register in a register file is a private register or a global register, each bit in the 64-bit special register corresponding to one of said registers in said register file.

28. (Original) The computer processing architecture as recited in claim 26, wherein said program counter register is a global register.

29. (Original) In a processing core comprising a processing pipeline having N-number of processing paths, each of said processing paths for processing instructions on M-bit data words, and one or more register files having Q-number of registers, said Q-number of registers being M-bits wide, a method for jumping from one location in a program to another location in a program, comprising the steps of:

storing a current program counter value in a program counter register, which is one of said Q-number of register in at least one of said one or more register files; and

adding a value to said current program counter value stored in said program counter register using a standard add operation.

30. (Original) In a processing core comprising a processing pipeline having N-number of processing paths, each of said processing paths for processing instructions on M-bit data words, and one or more register files having Q-number of registers, said Q-number of registers being M-bits wide, a method for calculating a memory address, comprising the steps of:

storing a current program counter value in a program counter register which is one of said Q-number of register in at least one of said one or more register files; and

adding a value to said current program counter value stored in said program counter register using a standard add operation.